
SSSSSSSSSSS YYY YYY SSSSSSSSSSS
SSSSSSSSSSS YYY YYY SSSSSSSSSSS
SSSSSSSSSSS YYY YYY SSSSSSSSSSS
SSS YYY YYY SSS
SSS YYY YYY SSS
SSS YYY YYY SSS
SSS YYY YYY SSS
SSS YYY YYY SSS
SSS YYY YYY SSS
SSSSSSSSS YYY SSSSSSSSS
SSSSSSSSS YYY SSSSSSSSS
SSSSSSSSS YYY SSSSSSSSS
SSS YYY SSS
SSS YYY SSS
SSS YYY SSS
SSS YYY SSS
SSS YYY SSS
SSS YYY SSS
SSSSSSSSSSS YYY SSSSSSSSSSS
SSSSSSSSSSS YYY SSSSSSSSSSS
SSSSSSSSSSS YYY SSSSSSSSSSS

```
  SSSSSSSS  YY      YY   SSSSSSSS  EEEEEEEEEE  RRRRRRRR     AAAAAA    PPPPPPPP     AAAAAA    TTTTTTTTTT
  SSSSSSSS  YY      YY   SSSSSSSS  EEEEEEEEEE  RRRRRRRR     AAAAAA    PPPPPPPP     AAAAAA    TTTTTTTTTT
SS          YY      YY  SS         EE          RR      RR  AA    AA  PP      PP  AA    AA      TT
SS          YY      YY  SS         EE          RR      RR  AA    AA  PP      PP  AA    AA      TT
SS            YY  YY    SS         EE          RR      RR  AA    AA  PP      PP  AA    AA      TT
SS            YY  YY    SS         EE          RR      RR  AA    AA  PP      PP  AA    AA      TT
  SSSSSS        YY        SSSSSS   EEEEEEEE    RRRRRRRR   AA    AA  PPPPPPPP    AA    AA      TT
  SSSSSS        YY        SSSSSS   EEEEEEEE    RRRRRRRR   AA    AA  PPPPPPPP    AA    AA      TT
        SS      YY              SS  EE          RR  RR     AAAAAAAAAA  PP          AAAAAAAAAA      TT
        SS      YY              SS  EE          RR  RR     AAAAAAAAAA  PP          AAAAAAAAAA      TT
        SS      YY              SS  EE          RR    RR   AA    AA    PP          AA    AA        TT          ....
        SS      YY              SS  EE          RR    RR   AA    AA    PP          AA    AA        TT          ....
SSSSSSSS        YY      SSSSSSSS    EEEEEEEEEE  RR      RR AA    AA    PP          AA    AA        TT          ....
SSSSSSSS        YY      SSSSSSSS    EEEEEEEEEE  RR      RR AA    AA    PP          AA    AA        TT          ....

LL              IIIIII     SSSSSSSS
LL              IIIIII     SSSSSSSS
LL                II      SS
LL                II      SS
LL                II      SS
LL                II      SS
LL                II        SSSSSS
LL                II        SSSSSS
LL                II              SS
LL                II              SS
LL                II              SS
LL                II              SS
LLLLLLLLLL      IIIIII    SSSSSSSS
LLLLLLLLLL      IIIIII    SSSSSSSS
```

Table of contents

```
0000     1          .TITLE  SYSERAPAT - Generate a security erase pattern
0000     2          .IDENT  'V04-000'
0000     3 ;
0000     4 ;*****************************************************************************
0000     5 ;*                                                                           *
0000     6 ;*  COPYRIGHT (c) 1978, 1980, 1982, 1984 BY                                  *
0000     7 ;*  DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASSACHUSETTS.                   *
0000     8 ;*  ALL RIGHTS RESERVED.                                                     *
0000     9 ;*                                                                           *
0000    10 ;*  THIS SOFTWARE IS FURNISHED UNDER A LICENSE AND MAY BE USED AND COPIED    *
0000    11 ;*  ONLY IN  ACCORDANCE WITH  THE  TERMS  OF  SUCH  LICENSE  AND WITH THE   *
0000    12 ;*  INCLUSION OF THE ABOVE COPYRIGHT NOTICE. THIS SOFTWARE OR  ANY  OTHER    *
0000    13 ;*  COPIES THEREOF MAY NOT BE PROVIDED OR OTHERWISE MADE AVAILABLE TO ANY    *
0000    14 ;*  OTHER PERSON.  NO TITLE TO AND OWNERSHIP OF  THE  SOFTWARE IS  HEREBY    *
0000    15 ;*  TRANSFERRED.                                                             *
0000    16 ;*                                                                           *
0000    17 ;*  THE INFORMATION IN THIS SOFTWARE IS  SUBJECT TO CHANGE WITHOUT NOTICE    *
0000    18 ;*  AND  SHOULD  NOT  BE  CONSTRUED AS  A COMMITMENT BY DIGITAL EQUIPMENT   *
0000    19 ;*  CORPORATION.                                                             *
0000    20 ;*                                                                           *
0000    21 ;*  DIGITAL ASSUMES NO RESPONSIBILITY FOR THE USE  OR  RELIABILITY OF ITS    *
0000    22 ;*  SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DIGITAL.                  *
0000    23 ;*                                                                           *
0000    24 ;*                                                                           *
0000    25 ;*****************************************************************************
0000    26
0000    27 ;++
0000    28 ; FACILITY: VMS Executive, System services.
0000    29 ;
0000    30 ; ABSTRACT:
0000    31 ;
0000    32 ;       Generate and return a security erase pattern.  This code
0000    33 ;       is more or less a place holder for a user written routine to
0000    34 ;       accomplish the same function.  The erase pattern returned by
0000    35 ;       this routine will always be zero.
0000    36 ;
0000    37 ;
0000    38 ; ENVIRONMENT:
0000    39 ;
0000    40 ;       Kernel Mode
0000    41 ;
0000    42 ; AUTHOR:
0000    43 ;
0000    44 ;       Steven T. Jeffreys
0000    45 ;
0000    46 ; CREATION DATE:
0000    47 ;
0000    48 ;        24-September-1982
0000    49 ;
0000    50 ; MODIFIED BY:
0000    51 ;
0000    52 ;       V03-001 STJ3054          Steven T. Jeffreys,          21-Jan-1983
0000    53 ;                Removed EXE$ERAPAT_DEF definition.
0000    54 ;--
0000    55 ;
```

```
                        0000     57          .SBTTL  Declarations
                        0000     58          $ERADEF                         ; Define erase type codes
                        0000     59          $PSLDEF                         ; Define PSL fields
                        0000     60          $SSDEF                          ; Define status codes
                        0000     61
                        0000     62 ;
                        0000     63 ; Equated symbols:
                        0000     64 ;
                        0000     65
00000004                0000     66          TYPE    = 4                     ; Offset to TYPE parameter   (value)
00000008                0000     67          COUNT   = 8                     ; Offset to COUNT parameter  (value)
0000000C                0000     68          PATADR  = 12                    ; Offset to PATADR parameter (address)
                        0000     69
00000001                0000     70          MAXCOUNT= 1                     ; Maximum count (erase 1 time)
```

```
                                                H 5
                                                .SBTTL  Entry vector
                                                ;+
                                                ; The following vectors are used by the various pieces of the system
                                                ; to access the erase pattern generator.  The vector EXE$ERAPAT is
                                                ; used by the change mode dispatcher in response to a user calling the
                                                ; $ERAPAT system service.  This vector then jumps to the actual dispatch
                                                ; vector, EXE$ERAPAT_VEC, which in turn will jump to erase pattern
                                                ; generator code.  This level of indirection is necessary because the
                                                ; change mode dispatch vector must be in close proximity to the change
                                                ; mode dispatcher, which implies that it must be in a read-only psect.
                                                ; The actual dispatch vector, EXE$ERAPAT_VEC, must be in a writable
                                                ; psect so that the contents of the vector may be changed.
                                                ;
                                                ; The longword SGN$GL_LOADFLAGS is a bit vector used to indicate which
                                                ; pieces of the loadable pieces of the EXEC should be loaded at system
                                                ; boot time.  If a user specified erase pattern generator routine is
                                                ; present in the system, the bit SGN$V_LOADERAPT will be set to 1.
                                                ; This fact can be used to the advantage of the EXEC to avoid the overhead
                                                ; of having to call the default erase pattern generator, since it always
                                                ; returns a zero, and is a one-step erase function.
                                                ;
                                                ; The vector address the user must specify to load the code is represented
                                                ; by the symbol EXE$ERAPAT_VEC.
                                                ;
                                                ;-

                                00000000        .PSECT  AEXENONPAGED            ; Nonpaged UR access only
                                                EXE$ERAPAT::                    ; Entry point from change-mode dispat.
                                0000            .WORD   0                       ; Register save mask (none saved)
00000000'9F     17  0002                        JMP     @#EXE$ERAPAT_VEC        ; Jump to the dispatch vector
                    0008
                                00000000        .PSECT  $$$500                  ; The vector must be nonpaged and URKW
                                                EXE$ERAPAT_VEC::                ; Quick access entry point
00000000'9F     17  0000                        JMP     @#EXE$ERAPAT_RTN        ; Vector to default routine
```

```
                     0006   107          .SBTTL  Main routine
                     0006   108 ;++
                     0006   109 ; $ERAPAT
                     0006   110 ;
                     0006   111 ; Functional description:
                     0006   112 ;
                     0006   113 ;       In order to perform a multi-step security erase, the caller repeatedly
                     0006   114 ;       calls this service, each time incrementing the iteration count.  After
                     0006   115 ;       each call, the erase pattern returned is written in the user supplied
                     0006   116 ;       area.  (The user is responsible for propagating that pattern throughout
                     0006   117 ;       memory, disk, tape, etc.)  When the service returns SS$_NOTRAN in R0,
                     0006   118 ;       the security erase operation is complete.
                     0006   119 ;
                     0006   120 ;       This simple routine will always return an erase pattern of 0.  It is
                     0006   121 ;       up to the system mangager to provide a specialized load algorithm.
                     0006   122 ;
                     0006   123 ; Calling sequence:
                     0006   124 ;
                     0006   125 ;       This routine should be called via a CALLS/G to EXE$ERAPAT.
                     0006   126 ;
                     0006   127 ; Input:
                     0006   128 ;
                     0006   129 ;       TYPE(AP)        : Security erase type.  The legal types are
                     0006   130 ;                               ERA$K_MEMORY : main memory
                     0006   131 ;                               ERA$K_DISK   : disk storage
                     0006   132 ;                               ERA$K_TAPE   : tape storage
                     0006   133 ;
                     0006   134 ;       COUNT(AP)       : Iteration count.  The service should be called
                     0006   135 ;                         the first time with the value 1, then 2, etc.,
                     0006   136 ;                         until the status SS$_NOTRAN is returned.  The
                     0006   137 ;                         local symbol MAXCOUNT defines how many times this
                     0006   138 ;                         happens.
                     0006   139 ;
                     0006   140 ; Output:
                     0006   141 ;
                     0006   142 ;       PATADR(AP)      : Address of a longword into which the security
                     0006   143 ;                         erase pattern is to be written.
                     0006   144 ;
                     0006   145 ; Routine value:
                     0006   146 ;
                     0006   147 ;       R0 = SS$_ACCVIO         : pattern output area not accessible
                     0006   148 ;            SS$_BADPARAM       : invalid security type code
                     0006   149 ;            SS$_NORMAL         : normal successful completion
                     0006   150 ;            SS$_NOTRAN         : security erase complete
                     0006   151 ;--
                     0006   152
                 00000000   153          .PSECT  Y$EXEPAGED                      ; This code is pageable
                     0000   154
                     0000   155 EXE$ERAPAT_RTN::                                 ; $ERAPAT code
   50    14    3C    0000   156          MOVZWL  #SS$_BADPARAM,R0                ; Assume bad parameter value
51    04 AC    D0    0003   157          MOVL    TYPE(AP),R1                     ; Get the type code
                     0007   158          ASSUME  ERA$K_MINTYPE EQ 1              ; This must be true if BLEQ is to work
         27    15    0007   159          BLEQ    69$                             ; Branch if type code too small
   51    03    D1    0009   160          CMPL    #ERA$K_MAXTYPE,R1               ; Is the type code too big?
         22    19    000C   161          BLSS    69$                             ; Branch if yes
51    08 AC    D0    000E   162          MOVL    COUNT(AP),R1                    ; Get the count
         1C    15    0012   163          BLEQ    69$                             ; Branch if too small
```

```
                50   0629 8F    3C    0014   164          MOVZWL  #SS$_NOTRAN,R0          ; Assume count too big
                     51   01    D1    0019   165          CMPL    #MAXCOUNT,R1           ; Are we done?
                          12    19    001C   166          BLSS    69$                    ; If less, then yes
                     50   0C    3C    001E   167          MOVZWL  #SS$_ACCVIO,R0         ; Assume access violation
                51   0C AC      D0    0021   168          MOVL    PATADR(AP),R1          ; Get address of user buffer
                                   0025   169          IFNOWRT #4,(R1),69$            ; Branch if no write access
                          61    D4    0028   170          CLRL    (R1)                   ; Return the erase pattern
                     50   01    3C    002B   171          MOVZWL  #SS$_NORMAL,R0         ; Set success status
                                04    002E   172 69$:      RET                            ; Return
                                      0031   173
                                      0031   174          .END
```

```
COUNT                   = 00000008
ERA$K_MAXTYPE           = 00000003
ERA$K_MINTYPE           = 00000001
EXE$ERAPAT                00000000 RG    02
EXE$ERAPAT_RTN            00000000 RG    04
EXE$ERAPAT_VEC            00000000 RG    03
MAXCOUNT                = 00000001
PATADR                  = 0000000C
SS$_ACCVIO              = 0000000C
SS$_BADPARAM            = 00000014
SS$_NORMAL              = 00000001
SS$_NOTRAN              = 00000629
TYPE                    = 00000004
```

Psect synopsis

PSECT name	Allocation	PSECT No.	Attributes
. ABS .	00000000 (0.)	00 (0.)	NOPIC USR CON ABS LCL NOSHR NOEXE NORD NOWRT NOVEC BYTE
ABS	00000000 (0.)	01 (1.)	NOPIC USR CON ABS LCL NOSHR EXE RD WRT NOVEC BYTE
AEXENONPAGED	00000008 (8.)	02 (2.)	NOPIC USR CON REL LCL NOSHR EXE RD WRT NOVEC BYTE
$$$500	00000006 (6.)	03 (3.)	NOPIC USR CON REL LCL NOSHR EXE RD WRT NOVEC BYTE
Y$EXEPAGED	00000031 (49.)	04 (4.)	NOPIC USR CON REL LCL NOSHR EXE RD WRT NOVEC BYTE

Performance indicators

Phase	Page faults	CPU Time	Elapsed Time
Initialization	35	00:00:00.08	00:00:00.75
Command processing	131	00:00:00.57	00:00:04.17
Pass 1	207	00:00:04.38	00:00:14.66
Symbol table sort	0	00:00:00.68	00:00:01.85
Pass 2	52	00:00:00.84	00:00:02.83
Symbol table output	3	00:00:00.03	00:00:00.03
Psect synopsis output	2	00:00:00.04	00:00:00.32
Cross-reference output	0	00:00:00.00	00:00:00.00
Assembler run totals	432	00:00:06.62	00:00:24.67

```
The working set limit was 1200 pages.
23328 bytes (46 pages) of virtual memory were used to buffer the intermediate code.
There were 30 pages of symbol table space allocated to hold 460 non-local and 1 local symbols.
174 source lines were read in Pass 1, producing 17 object records in Pass 2.
12 pages of virtual memory were used to define 11 macros.
```

```
                                +----------------------------+
                                ! Macro library statistics !
                                +----------------------------+

Macro library name                          Macros defined
------------------                          --------------
_$255$DUA28:[SYS.OBJ]LIB.MLB;1                       1
_$255$DUA28:[SYSLIB]STARLET.MLB;2                    7
TOTALS (all libraries)                               8

533 GETS were required to define 8 macros.

There were no errors, warnings or information messages.

MACRO/LIS=LIS$:SYSERAPAT/OBJ=OBJ$:SYSERAPAT MSRC$:SYSERAPAT/UPDATE=(ENH$:SYSERAPAT)+EXECML$/LIB
```

SYSGETJPI
LIS

SYSERAPAT
LIS

SYSFAO
LIS

SYSGETDVI
LIS

SYSEXIT
LIS

SYSEVTSRV
LIS

SYSFORCEX
LIS